VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky

BAKALÁŘSKÁ PRÁCE

2010 David Balcárek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Přestavba a optimalizace metropolitní sítě

Redesign and Optimalization of Metropolitan Network

2010 David Balcárek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **David Balcárek**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Přestavba a optimalizace metropolitní sítě
Redesign and Optimalization of Metropolitan Network

Zásady pro vypracování:

Infrastruktura sítě v lokalitě Bohumín je v současné době realizována na druhé vrstvě ISO-OSI modelu. Se zvyšováním počtu klientských stanic je tento stav dále nevyhovující. Je potřeba síť rozdělit do podsítí propojených na třetí vrstvě a tuto dále optimalizovat.

1. Navrhněte novou strukturu sítě založenou na prvcích 3. vrstvy OSI modelu. Zhodnoťte více možností návrhu a vyberte nejlepší z nich.
2. Směrování realizujte s použitím protokolu OSPF. Zvažte rozdělení sítě do OSPF oblastí.
3. Implementujte vhodný systém rate-limitingu.
4. Připravte skripty pro podporu administrace.
5. Navrhněte další možnosti pro celkové vylepšení sítě.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Petr Grygárek, Ph.D.**

Datum zadání: 30.11.2008

Datum odevzdání: 07.05.2010

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

doc. Dr.Ing. Eduard Sojka
vedoucí katedry

prof. Ing. Ivo Vondrák, CSc.
děkan fakulty

Prohlašuji, že jsem bakalářskou práci na téma přestavba a optimalizace metropolitní sítě vypracoval samostatně s použitím odborné literatury a pramenů, uvedených na seznamu, který tvoří přílohu této práce.

V Ostravě …….

David Balcárek

Rád bych poděkoval vedoucím práce Ing. Petru Grygárkovi Ph.D. a Ing. Tomáši Kučerovi, za odborné vedení, rady a připomínky při zpracování bakalářské práce. Dále bych poděkoval bývalým i současným pracovníkům projektu CZFree Bohumín, kteří přispěli svými připomínkami, jsou to Bc. Václav Bortlík, Bc. Darek Červeňák a Radomil Mikolanda.

Abstrakt

Bakalářská práce popisuje návrh metropolitní sítě na 3. vrstvě OSI modelu, neboť stávající model realizován na 2. vrstvě je nevyhovující. Jedná se o rekonstrukci stávající radiové sítě. S realizací sítě na 3. vrstvě je spjata i vhodná implementace směrování pomocí protokolu OSPF a rate-limitingu. Součásti této práce jsou skripty pro podporu administrace a jednoduchý návod, jak vytvářet vlastní skripty v operačním systému RouterOS od společnosti Mikrotik.

Klíčová slova: OSPF, rate-limiting, RouterOS, Mikrotik

Abstract

My bachelor's thesis describes the design of the metropolitan network on the 3rd layer of the OSI model, because the current model implemented on the 2nd layer is unsatisfactory. It is a reconstruction of the existing radio network. The appropriate implementation of routing by means of the OSPF protocol and rate-limiting is also connected with the implementation on the 3rd layer. This work also includes scripts to support the administration and simple instructions on how to create own scripts in the RouterOS operating system from Mikrotik company.

Key words: OSPF, rate-limiting, RouterOS, Mikrotik

Seznam použitých symbolů a zkratek

ABR	- Area Border Router
ARP	- Address Resolution Protocol
ASBR	- Autonomous System Boundary Router
BFIFO	- Bytes First-In First-Out
BGP	- Border Gateway Protocol
CLI	- Command-Line Interface
DVA	- Distance Vector Algorithms
FIFO	- First-In First-Out
FTP	- File Transfer Protocol
GB	- 1 gigabyte = 1073741824 bytů
HTB	- Hierarchical Token Bucket
IP	- Internet Protocol
IR	- Internal Router
ISDN	- Integrated Services Digital Network
kB	- 1 kilobyte = 1024 bytů
kb	- 1 kilobit = 1000 bitů
LSA	- Link State Algorithms
MAC	- Media Access Control
MB	- 1 megabyte = 1048576 bytů
Mb	- 1 megabit = 1000000 bitů
MD5	- Message-Digest algorithm 5
NAT	- Network Address Translation
NSSA	- Not So Stubby Area
OSPF	- Open Shortest Path First
PC	- Personal Computer

PCQ	- Per Connection Queue
PFIFO	- Packets First-In First-Out
PoE	- Power over Ethernet
RED	- Random Early Detect
RIP	-Routing Information Protocol
SFQ	- Stochastic Fairness Queuing
SMTP	- Simple Mail Transfer Protocol
SNMP	- Simple Network Management Protocol
SSH	- Secure Shell
VLAN	- Virtuální LAN
VoIP	- Voice over IP
VPN	- Virtual Private Network
VRRP	- Virtual Router Redundancy Protocol
WEP	- Wired Equivalent Privacy
Wi-Fi	- Wireless Fidelity
WPA	- Wi-Fi Protected Access
WPA2	- Wi-Fi Protected Access 2

Obsah

1 Úvod

Před několika lety jsme se s několika přáteli rozhodli propojit naše počítače do jedné sítě. Pro sdílení dat a hraní her byl Internet v té době velmi pomalý (56 kb/s pomocí modemu, 128 kb/s pomocí ISDN modemu) a poměrně drahý. Protože ne všichni bydlíme ve stejném domě, bylo jedinou možností použít bezdrátová zařízení. Zaujala nás radiová zařízení operující v kmitočtovém pásmu 2,4 GHz, dnes veřejně známá pod zkratkou Wi-Fi. V tomto pásmu jsou v ČR používaný dva standardy a to IEEE 802.11b a IEEE 802.11g. Liší se v použití fyzické vrstvy. Dnes je již také běžný standard IEEE 802.11h, který je obdobou IEEE 802.11g, ale operuje v kmitočtovém pásmu 5 GHz.

Po několika týdnech provozu bezdrátové sítě jsme zakoupili konektivitu od poskytovatele Internetu. Konektivitu nám rovnoměrně rozděloval jednoduchý směrovač. Bylo to staré PC (Pentium 133 MHz, 32 MB RAM) s operačním systémem na bázi Linuxu. Tento operační systém nevyžadoval použití harddisku, tak jsme jej umístili na disketu o kapacitě 1,44 MB, protože vždy při načítání počítače se načetl celý do operační paměti.

Při hledání patřičného softwaru na omezování rychlosti internetu koncovým uživatelům, jsme narazili na projekt CZFree, komunitu lidí, kteří v rámci občanských sdružení budují a spravují regionální sítě. Projekt nás zaujal a přidali jsme se do něj také. Dostali jsme přidělený privátní rozsah IP adres pro budoucí propojení s jinou sítí z tohoto projektu. Nečekali jsme, že zájem o připojení k Internetu bude tak velký, proto jsme zůstali u toho nejjednoduššího modelu počítačové sítě. To znamená jeden segment IP adres, pouze jeden směrovač a to brána do Internetu.

Cílem mé práce je navrhnout a popsat síť realizovanou na 3. vrstvě OSI modelu. Jedná se o přestavbu již funkční radiové sítě v Bohumíně. Nejprve vysvětlím, jak síť vypadala, jaké měla problémy a co bylo důvodem k přestavbě. Po vybrání vhodného směrovače se zaměřím na směrovací protokoly, které podporuje. Porovnám je a vysvětlím, proč směrovací protokol OSPF je nejlepší volba. Po navrhnutí OSPF oblastí rozdělím nám přiřazený rozsah IP adres na menší podsítě. Když už je celá síť navržena, vysvětlím jaké možnosti rate-limitingu nám operační systém směrovače dovoluje. Po zvážení možností popíšu realizaci vybraného mechanismus rate-limitingu. Nakonec se pokusím vysvětlit jak vytvářet skripty pro operační systém směrovače a uvedu několik příkladů.

2 Základní informace

Síť má v tuto chvíli okolo 380 uživatelů. Jejich počet pomalu stoupá. Přístupových bodů pro připojení uživatelů je 24. Celkový počet aktivních prvků v síti je okolo 500 z toho na zařízení uživatelů připadá přibližně 400. Celkovou konektivitu do internetu máme garantovaných 25 Mb/s / 25 Mb/s. Uživatelé mají tarif 3 Mb/s / 1 Mb/s s agregací 1:10. Reálné hodnoty, které jsme naměřili u uživatelů, se pohybovaly mezi 1 Mb/s – 2 Mb/s download (tab. 1) a 100 kb – 500 kb upload (proběhlo celkem 200 měření u 50 uživatelů v různou denní dobu).

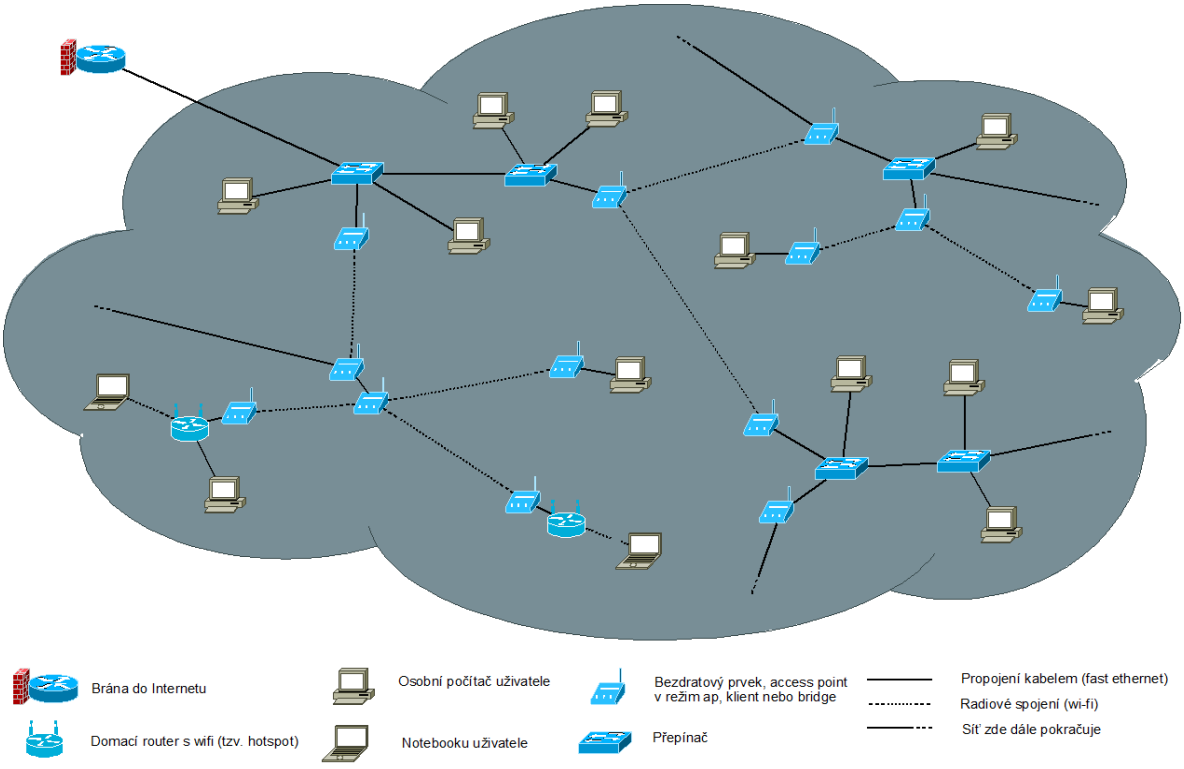
Pondělí:		**naměřené hodnoty**	
čas měření	**minimální**	**maximální**	**průměr**
mezi 9:00 - 10:00	1,3 Mb/s	1,9 Mb/s	1,7 Mb/s
mezi 19:00 - 20:00	1 Mb/s	1,5 Mb/s	1,2 Mb/s
Čtvrtek:		**naměřené hodnoty**	
čas měření	**minimální**	**maximální**	**průměr**
mezi 7:00 - 8:00	1,4 Mb/s	2 Mb/s	1,8 Mb/s
mezi 21:00 - 22:00	1 Mb/s	1,4 Mb/s	1,3 Mb/s

Tabulka 1: Měření rychlosti download u uživatelů.
Měření rychlosti proběhlo pomocí ukazatele rychlosti na serveru www.dsl.cz na počítačích jednotlivých uživatelů. V každou denní dobu v tabulce bylo provedeno 1 měření u 50 uživatelů. Bohužel v době pořizování měření server www.dsl.cz neposkytoval měření uploadu.

Při přidělení IP adres nám bylo doporučeno, abychom již od počátku rozdělili rozsah na menší části a směrovali. Jak již jsem psal výše, nečekali jsme, že se síť tak rozroste, proto jsme původně chtěli použit celý segment IP adres. Nakonec jsme se rozhodli pro rozdělení na dva stejně velké segmenty pro případnou realizaci směrování. První segment jsme nechali volný a celou síť jsme lokalizovali do druhého segmentu. Takže z celého segmentu 10.157.0.0/16 používáme nyní 10.157.128.0/17 a zbylá část 10.157.0.0/17 byla ponechána pro případ přestavby.

Do sítě Internet přistupujeme přes směrovač s operačním systémem Linux. Na tomto směrovači je spuštěn také firewall (obr. 1). Celý rozsah 10.157.128.0/16 je privátní a proto je na tomto směrovači spuštěna služba NAT. Tato služba je nastavena tak, že pod jednou veřejnou IP adresou, nám přidělenou od poskytovatele, je skryta celá síť s privátním rozsahem. V případě, že uživatel potřeboval veřejnou adresu, tak jsme ji realizovali překladem adres 1:1, kdy k jedné veřejné IP adrese patřila právě jedna privátní. Týkalo se to hlavně firemních zákazníku, kteří potřebovali mít přístup na kamerový systém popřípadě jiný server i z Internetu.

Nyní na naší síti můžeme nalézt různé typy zařízení různých výrobců. Mezi ty nejpoužívanější patří výrobky firmy OvisLink, TP-Link. Všechny tyto zařízení jsou v režimu bridge, tedy jsou mezi rozhraními průchozí a mají jen jednu IP adresu pro management. Výjimkou jsou zařízení u koncových uživatelů. Hodně lidi si v poslední době pořizuje svůj domácí hotspot, který má zapnutou funkci NAT tak jako náš směrovač do internetu. Díky tomu mohou uživatele se svým notebookem být připojení kdekoliv doma bez nutností použití kabelu.



Obr. 1: Náčrt části sítě (pro ilustraci přibližně 1/20 celé sítě).

2.1 Problémy rozvíjející se sítě

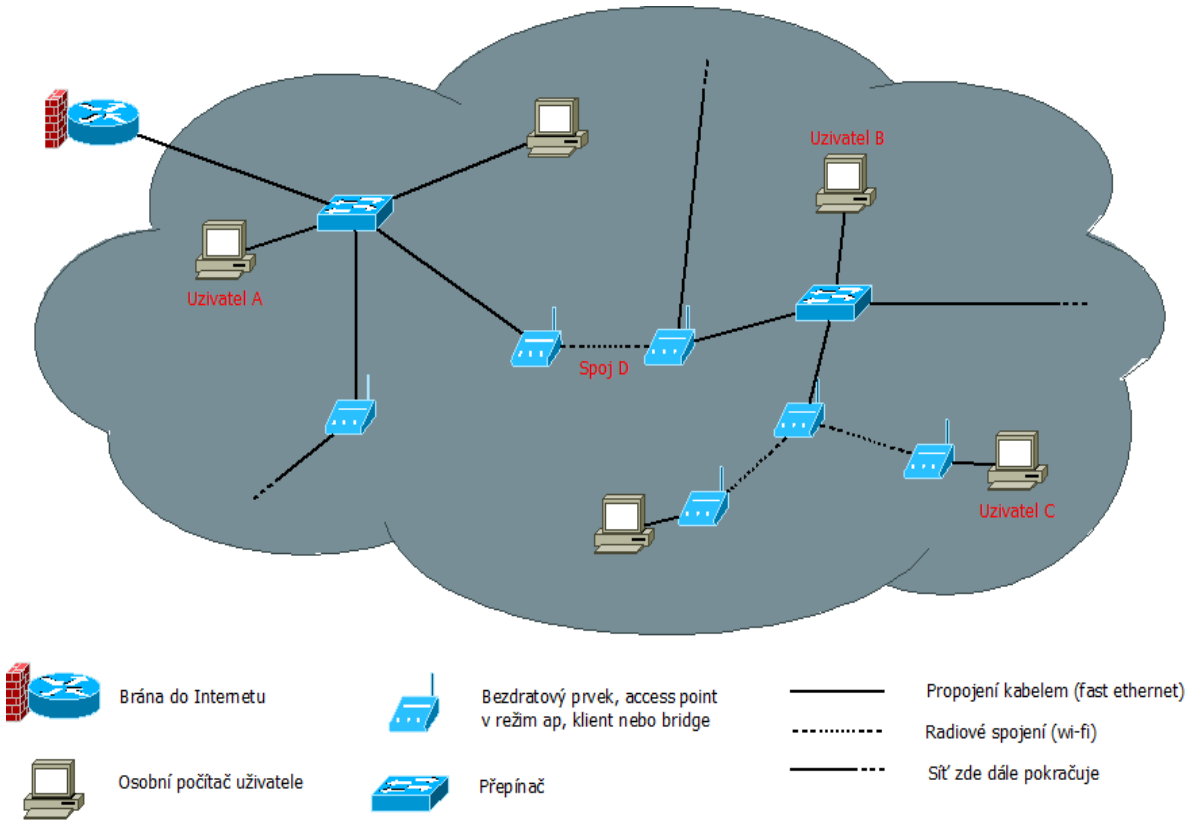
Postupem času se ukazuje, že nynější návrh sítě je s rostoucím počtem uživatelů nevyhovující. Roste počet broadcastů, které se pohybují sítí (tab. 2). Broadcast je paket, jehož cílovou adresou je celý segment sítě. Z toho vyplývá, že čím větší segment sítě, tím větší počet těchto paketů se pohybuje v síti. Tento typ paketu využívá ARP, některé komunikační programy, hry, viry a spousta dalších služeb. Pod tím si můžeme představit například samovolné rozesílání virové nákazy všem počítačům v síti. Tento provoz bohužel nemůžeme zakázat (je potřebný pro některé důležité služby) ani ignorovat, protože virová nákaza může způsobovat problémy v komunikaci v síti. Vhodným řešením je rozdělení celé sítě na menší segmenty adres, kde každý uživatel bude mít právě jeden menší segment adres. V takovémto

případě ať už chtěné, tak nechtěné broadcasty uživatele zůstanou v jeho segmentu a nebudou žádným způsobem ovlivňovat jiné uživatele na síti.

Pondělí:				
čas měření	**počet paketů**	**pakety/sekundu**	**z toho broadcastů**	
8:00 - 9:00	22781493	6328	8352245	36,7%
19:00 - 20:00	48327216	13424	15543891	32,2%
Čtvrtek:				
čas měření	**počet paketů**	**pakety/sekundu**	**z toho broadcastů**	
8:00 - 9:00	28162345	7822	10854249	38,5%
19:00 - 20:00	45346624	12596	14875243	32,8%
Sobota:				
čas měření	**počet paketů**	**pakety/sekundu**	**z toho broadcastů**	
8:00 - 9:00	24843827	6901	7246147	29,2%
19:00 - 20:00	53167493	14768	13246987	24,9%

Tabulka 2: Měření počtu paketů za hodinu.
Pomocí programu WireShark byla hodinu odchytávána komunikace před směrovačem do Internetu. Z výsledného záznamu jsem odfiltroval pakety, jejíž cílová adresa byla IP adresa broadcastu.

Dalším problémem, který se časem objevil, je kopírování dat mezi uživateli. S rostoucími rychlostmi do Internetu, jsou linky stále více využívány. Čím dál častěji dochází k situaci, kdy uživatelům poklesne rychlost Internetu i na pár kB, protože dva účastníci v síti si posílají větší objem dat (obvykle několik GB). Je to zapříčiněno tím, že omezujeme rychlost připojení do Internetu, nikoliv rychlost přístupu uživatele k síti. Vysvětlení pomocí obrázku číslo 2. Uživatel A a uživatel B kopírují mezi sebou cca 10 GB. Nejužším místem na trase mezi nimi je bezdrátový spoj D, který propustí ve skutečnosti přibližně 8 Mb download a 8 Mb upload, což je několikrát méně než samotný Fast Ethertnet. V tu samou chvíli přistupuje do Internetu uživatel C. Během doby kopírování uživatelů A a B, uživatel C čeká delší dobu (až několik sekund) než se stránka začne načítat. Způsobuje to velké množství paketů, které se hromadí ve frontě na access pointech spoje D. Řešením je omezení rychlosti připojení uživatele do sítě. To můžeme provést záměnou stávajících switchů za switche managementovatelné nebo umístěním směrovače co nejblíže uživateli. Směrovač navíc může upřednostňovat provoz do Internetu před provozem v samotné síti a tím poskytnout maximální rychlost přenosu dat v síti a zároveň zajistit přístup uživatelům do Internetu.



Obr. 2: Příklad problému při kopírování dat mezi uživateli.

Třetí velkou nevýhodou je bezpečnost v tak rozsáhle síti, implementované do jednoho segmentu. Šifrovací klíče nejsou neprolomitelné a v případě útoku, je velmi těžké odhalit, odkud tento útok přichází. Toto se týká hlavně bezdrátových spojů, kde šifrování WEP je již nedostačující. Alternativou tohoto zabezpečení je například WPA či novější WPA2. V případě implementace směrovačů a rozdělení do segmentů je možné zjistit, odkud útočník do sítě přistupuje a reagovat na to. Samozřejmě nejde jen o ochranu vůči útoku z vnějšku. V jednom segmentu je možné odposlouchávat cizí komunikaci a tím získávat citlivá dat jiných uživatelů. Rozdělením na více segmentů tuto možnost zcela vymítíme. Pro realizaci je potřeba umístit směrovač co nejblíže uživateli a vytvořit pro něj segment s vlastním rozsahem adres. V místě, kde je více uživatelů připojeno Fast Ethernetem, je v hodné kromě směrovače použít i managementovatelný switch s funkcí VLAN.

Poslední důležitá věc, která není možná realizovat na nynější síti je vedení statistik přenesených dat na různých místech. Téměř žádné z námi používaných zařízení nepodporuje službu SNMP. Tato služba funguje jako server/klient. Server se nazývá SNMP agent. Pomocí SNMP klienta můžeme načíst počet přenesených dat, množství volné paměti, uptime zařízení a mnoho dalších informací, ale také můžeme nastavit některé atributy síťových prvků. Námi

používaná zařízení nepodporují také ani žádnou důmyslnější statistiku. Obvykle jediné co umí, je sčítat pakety nebo byte na rozhraních. Tyto informace jsou dostupné pouze přes webové rozhraní a strojově zpracovatelný vzdálený přístup k těmto informacím neumožňují. Těmito prostředky lze zmapovat provoz na různých místech sítě, konkrétně na směrovačích v síti, a podle nasbíraných údajů optimálně naplánovat posílení spojů.

3 Platforma a operační systém směrovače

Jelikož hledám zařízení pro umístění nejen na příhradový stožár, ale taky pro umístění k uživatelům na střechu, je třeba vyjmenovat, co musí a nemusí splňovat a podle toho vybrat jedno nebo více zařízení. Pro přestavbu budeme potřebovat výkonné, spolehlivé zařízení nejlépe s nízkou spotřebou a to vše za rozumnou cenu.

Jednou z možností je směrovač od společnosti Cisco a access point od společnosti OvisLink. Společnost Cisco již řadu let patří ke špičkovým výrobcům síťových prvků. Jejich zařízení se vyznačují kvalitou, spolehlivostí a velkou variabilitou nastavení. Bohužel všechny tyto klady jsou vyváženy vysokou cenou za pořízení. Veškerá zařízení od společnosti Cisco mají nainstalovaný operační systém IOS. Propojení s počítačem je obvykle přes konzolový port RJ45, samotný operační systém se obsluhuje přes CLI. Oproti tomu společnost OvisLink dělá cenově velmi přijatelné access pointy, které mají základní funkce. Podporují dynamická šifrování, například WPA. Jednoduchá obsluha, protože nastavení se provádí přes webový prohlížeč. Bohužel access point má jen jedno bezdrátové rozhraní, tak pro více bezdrátových spojů je třeba více access pointů. Několika zařízení sebou nese i problém s napájením. Ne vždy lze na střechu přivést 220 V, proto se z půdy vede napětí již z adapteru pomocí pasivního PoE. Více zařízení znamená i více natažených kabelů a zabere mnohem více místa, proto není tato kombinace vhodná pro umístění k uživatelům jako přístupový bod.

Další velmi populární volba je směrovač od společnosti Linksys. Linksys je již několik let divizí společnosti Cisco, což samo o sobě zaručuje kvalitu výrobků. Toto zařízení zastupuje funkci směrovače i bezdrátového access pointu. Výhody jsou jasné, jedno zařízení místo dvou. Z toho vyplývá nízká spotřeba a menší rozměry. Nastavení se provádí stejně jako u zařízení firmy OvisLink tedy přes webový prohlížeč. Jako jedinou nevýhodu tohoto řešení vidím, že pro menší retranslační stanici je potřeba více zařízení, jelikož každé má jen jedno bezdrátové rozhraní.

Nejzajímavější ze všech možností je použití směrovače postaveného na nějaké distribuci Linuxu. Má to nespočet výhod. Mezi ty nejdůležitější patří implementace libovolné služby, počínaje monitoringem konče Proxy serverem. Samozřejmě záleží na typu platformy, na kterou Linux nainstalujeme. V případě použití klasického PC dosáhneme velkého výkonu, takže implementace všech služeb bude bezproblémová. Nejsme ani příliš omezení počtem síťových karet. Ale PC je příliš velké a má velkou spotřebu, proto je umístěni k uživatelům opět nevhodné. Oproti tomu při implementaci na slabší platformu kompatibilní s x86 jsme omezeni počtem síťových karet, některé služby budou příliš náročné a tudíž nevhodné je využívat, ale dosáhneme nízké spotřeby a velikosti. Při použití kterékoliv platformy, je tady ještě problém

s implementací. Pro nastavení všech služeb a uvedení směrovače do chodu je potřeba větších znalostí. Jako podstatnou nevýhodu bych označil častou údržbu a s tím spojenou spolehlivost.

Poslední možnost, kterou tady zmíním, je platforma Routerboard od společnosti Mikrotik. Jak už z názvu vyplývá, jedná se o platformu primárně určenou pro směrování. Vyrábějí se různé modely, jež se liší kromě ceny hlavně ve výkonu a počtu síťových karet, které se do nich mohou nainstalovat. Pořizovací náklady jsou poměrně nízké. Výhodou je, že při tvorbě menšího přístupového bodu s celkem 4 bezdrátovými spoji, jednoduše osadíme platformu 4 bezdrátovými kartami a rozměry se nezvětší. Další výhodou je napájení, dokonce většina platforem má integrované pasivní PoE již v konektoru RJ45, není tedy třeba další místo na extraktor vedle zařízení. Má mnohem nižší spotřebu než několik zařízení (access point, směrovač …) zastupující stejnou funkci. Je vhodné i na místa s větší vzdáleností od přívodu napájení, protože obvykle napájecí rozpětí je od 8 V – 50 V a proto se dá snadno kompenzovat úbytek napětí s rostoucí vzdáleností. S tím je spjata i menší citlivost na výkyvy napětí v elektrické síti. Standardně zařízení dodávají s operačním systémem RouterOS, jenž je také od společnosti Mikrotik. Operační systém podporuje několik možností nastavení. První je přes webový prohlížeč. Je zde základní nastavení. Dále se RouterOS dá konfigurovat přes utilitu WinBox pro Windows, pomocí které se připojí přes IP nebo MAC adresu na zařízení a v GUI vše nastavit. A jako poslední možnost nastavení pomocí CLI a to buď přes Telnet, SSH a nebo sériový port. Jedinou nevýhodou tohoto softwaru jsou licence. Vyšší úroveň licence podporuje více aktivních spojení a stojí více peněz. Týká se to hlavně počtu aktivních VPN spojení.

Po pečlivém zvážení jsem vybral operační systém RouterOS od společnosti MikroTik. Jako platforma použijeme Routerboard, ale v případě potřeby většího výkonu můžeme MikroTik nainstalovat i na platformu x86 nebo x64.

V příloze 1 je zobrazen jeden z takových Routerboardů, který je vhodný pro tuto funkci. 3 sloty na bezdrátové karty je ideální počet, protože při umístění na stožár se nevleze vedle sebe víc jak 3 až 4 antény (větší zisk a menší úhel antény sebou obvykle přináší i větší rozměry).

4 Směrování

Pro přesun sítě z 2. vrstvy na 3. vrstvu OSI modelu rozdělíme náš rozsah IP adres na menší rozsahy a tyto podsítě následně spojíme směrovači do jednoho celku. Pro správnou funkčnost budeme muset správně vyplnit směrovací tabulku nebo použit nějaký ze směrovacích protokolů.

Statické směrování je ruční plnění směrovací tabulky oproti tomu u dynamického směrování se používají směrovací protokoly. Mezi ty nejpoužívanější patří RIP, OSPF, BGP.

4.1 Statické směrování

Je vhodné pro menší sítě o několika směrovačích. Důvodem je velké množství přidaných záznamů do každého směrovače. Po naplnění tabulky, vše funguje, ale při změně jednoho údaje je třeba opět změnu aktualizovat ve všech směrovačích v síti ručně. Pro jednu jedinou cestu lze použit výchozí brána, ta se definuje jako cílová adresa 0.0.0.0/0 a jako brána se nastaví IP adresa sousedního směrovače. U statické cesty lze někdy (podle použitého softwaru) nastavit i z jaké IP adresy budou pakety odcházet (jedno rozhraní může mít více IP adres ze stejného rozsahu), popřípadě typ kontroly brány (ping, ARP tabulka …) a mnoho dalších funkcí.

4.2 RIP (Routing Information Protocol)

Jeden z prvních směrovacích protokolů, vyvinut firmou Xerox v roce 1981. Využitá pro distribuci cest algoritmus DVA. Směrovače neznají topologii sítě, pouze rozhraní (adresy sousedů), přes která mají posílat pakety do jednotlivých sítí a vzdálenosti k těmto sítím (tzv. distanční vektory). Protokol je jednoduchý na implementaci, ale je omezen 16 přeskoky. Směrovač odesílá každých 30 s všem svým sousedům svou směrovací tabulku. Při přijetí směrovač rozhodne, zda zná lepší cestu k dané síti (podle počtu přeskoků). Pokud ano nahradí záznam ve své směrovací tabulce. Pokud ne zahodí jej. V případě, že je počet přeskoku označen za 16, je považován za neplatnou cestu (nedostupná síť).

4.4 OSPF (Open Shortest Path First)

Využívá pro distribuci cest algoritmus LSA. Provádí směrování na základě znalosti "stavu" jednotlivých linek sítě (funkčnost, cena). Tyto informace udržují v topologické databázi (všechny směrovače mají stejnou topologickou databázi). Každý směrovač počítá strom nejkratších cest ke všem ostatním směrovačům a k nim připojeným sítím pomocí Dijkstrova algoritmu. Protokol umožňuje rychlou konvergenci sítě reagující na jakékoli topologické změny

(např. porucha spoje) a nevede ke směrovacím smyčkám. Směrovač nejprve vypočte strom nejkratších cest a pak z něj vytvoří směrovací tabulku. Protokol podporuje hierarchické směrování (oblasti).

4.4 BGP (Border Gateway Protocol)

Je to vnější směrovací protokol pro distribucí cest mezi autonomními systémy (dnes má velké uplatnění v Internetu). Na rozdíl od vnitřních směrovacích protokolů (RIP, OSPF …) může mít každý autonomní systém jiný soubor kriterii pro určení optimální cesty (například rychlost, velikost autonomních systému po cestě, počet autonomních systému po cestě …).

Při výběru jaké nastavení zvolím, jsem postupně vylučoval možnosti dle reálných podmínek, které musí splňovat. Proto hned na začátek jsem vyloučil směrovací protokol BGP, jelikož budu spravovat jen jednu autonomní oblast. Dále v návrhu počítám s tím, že některé části sítě budou mít více možných cest k cíli (záloha linek v případě výpadku), a proto si myslím, že protokol RIP by příliš pomalu reagoval na změny v síti. RIP se, jak už bylo řečeno, obnovuje v pravidelných intervalech mezi sousedními směrovači, oproti tomu OSPF rozešle všem směrovačům změnu cesty. Každý směrovač si pak podle topologické tabulky přepočítá nejkratší cestu sám a vytvoří směrovací tabulku. Tyto záložní linky se objeví pouze na páteři celé sítě, protože téměř většina sítě je realizovaná bezdrátovými spoji a ty jsou čas od času zarušeny.

5 Rozdělení do OSPF oblasti

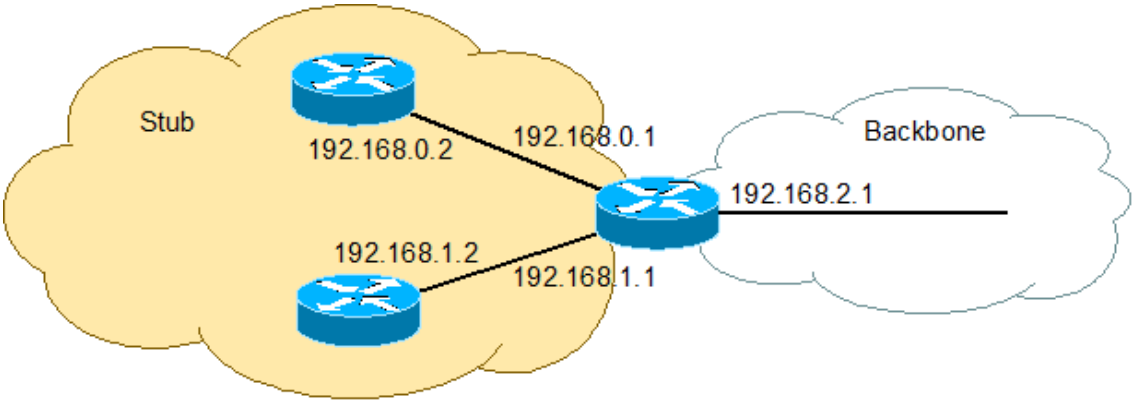
Při použití směrovacího protokolu OSPF na větší sítě může dojít zbytečnému zatížení směrovačů neustálými výpočty směrovacích tabulek. Jak příklad si vezmu naši síť. Po přestavbě odhaduji počet směrovačů okolo 50 až 100 kusů při stávajícím počtu uživatelů. Při realizaci pomocí směrovacího protokolu OSPF, by každá změna v topologii (například dočasné vypadnutí směrovače) se projevila změnou topologické tabulky ve všech směrovačích. Každý směrovač poté musí přepočítat celou směrovací tabulku znova. Při takhle rozsáhle síti by mohla být doba konvergence několik desítek sekund ne-li několik minut v závislosti na počtech směrovačů, jejich výkonnosti a množství segmentů. Tento výpočet musí provádět i směrovače na druhém konci města, kterým se výsledná směrovací tabulka nezmění. Proto pro větší efektivnost tohoto směrovacího protokolu byly do něj implementovány OSPF oblasti. Jedná se o logické rozčlenění sítě do uzavřených celků, které mají určité vlastnosti.

5.1 Typy oblastí implementující RouterOS

Společnost MikroTik zaimplementovala do OSPF balíčku 3 typy oblastí. Jsou to default, stub a nssa. Jejich rozdíly popíši níže.

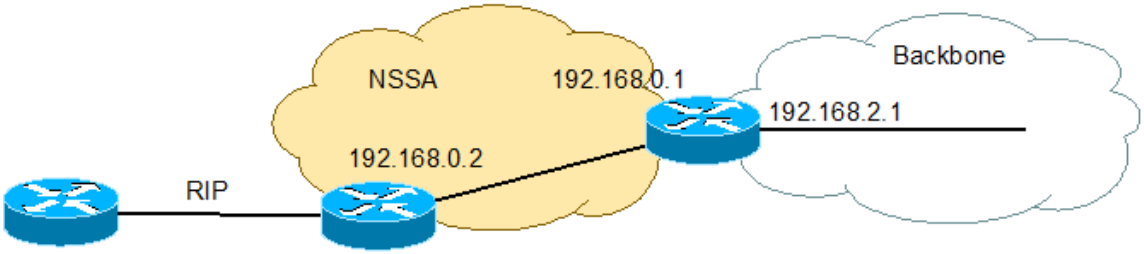
Prvním a základním typem OSPF oblasti je default. Tento typ oblasti se používá například u oblasti „backbone“, protože podle topologické tabulky se vytvoří cesty ke všem směrovačům. To znamená, že při každé změně se všem směrovačům v této oblasti zašlou změny do topologické tabulky. Poté směrovač přepočítá směrovací tabulku, která se nemusí vůbec změnit [1].

Dalším typem OSPF oblasti je oblast nazývána stub (obr. 3). Tento typ oblasti vznikl právě pro ošetření neustálého přepočtu směrovací tabulky. Tento typ oblasti zajistí, že mezi směrovači se předávají pouze cesty v dané oblasti a cesta ven ze sítě (do Internetu) je reprezentována výchozí cestou. Díky tomu se nemusí v směrovací tabulce uchovávat tak velké množství informací. Také se omezí počet přepočtů směrovací tabulky, díky nepotřeby přepočtu při výpadku některé z linek v jiné oblasti. Proto jsou veškeré výpočty jednodušší a výpočet směrovací tabulky rychlejší [1].



Obr. 3: OSPF oblast typu stub [2].

Poslední typ OSPF oblastí implementované v RouterOS se nazývá nssa. Tento typ je velmi podobný oblasti stub, ale umožňuje připojení i vnějších cest. Tím je myšlena propagace cest z jiného směrovacího protokolu (obr. 4). Přes tento typ oblasti jako jediný nelze vytvořit „virtual link“ s oblasti „backbone“ [1].



Obr. 4: OSPF oblast typu nssa [2].

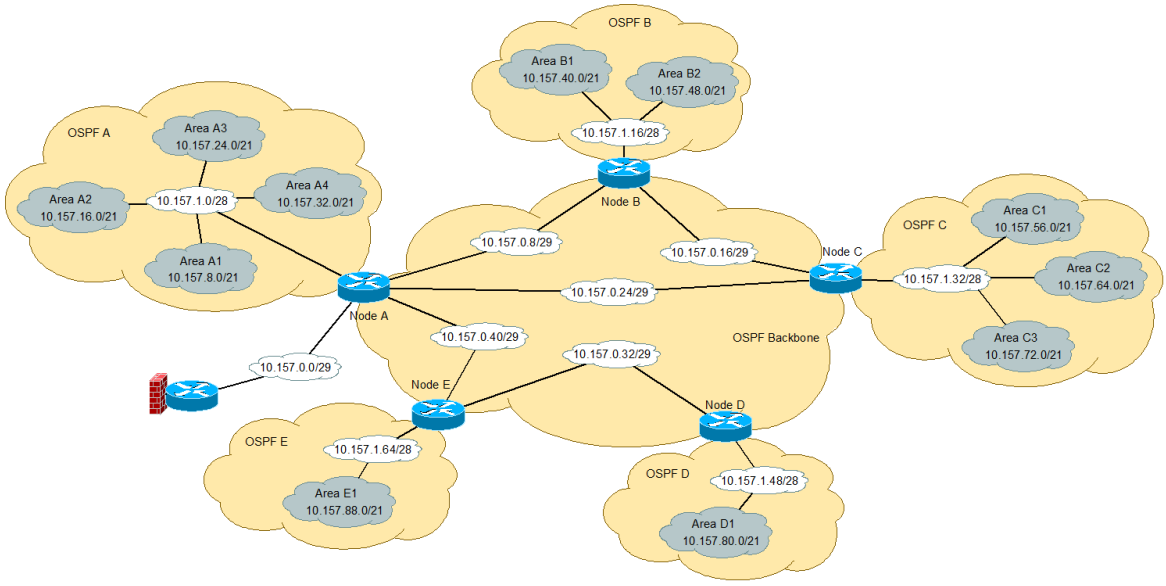
Virtual link je logické spojení oblastí. Má užití ve dvou případech, protože každá oblast musí být připojena do oblasti „backbone. Za prvé v případě, že nejde oblast fyzicky připojit k oblasti „backbone“. A za druhé v případě, že oblast „backbone“ je rozdělena na několik částí [1].

5.2 Návrh OSPF oblastí na naší síti

V mém návrhu oblastí směrovacího protokolu OSPF se opírám o geografické umístění zařízení. Všechny přístupové body připojené přímo nebo přes jiný přístupový bod na retranslační místo, budou jedna oblast. Retranslační místo je obvykle střecha věžového domu se stožárem. Takových to míst máme několik a jejich výhodou je možnost umístění většího množství antén.

Vytvořím oblast „backbone“, která bude typu default do níž budou připojeny zbylé oblasti. Všechny ostatní oblasti budou typu stub, protože budou koncové. Nepředpokládáme u nich připojení vnějších cest a propojení s „backbone“ bude vždy právě jedním směrovačem.

V oblasti „backbone“ bude jeden ASBR (takto označujeme směrovač zprostředkující cestu z OSPF oblasti „backbone“ k hlavní bráně do Internetu) a ostatní budou ABR (takto nazýváme směrovač, který je fyzicky připojen do více oblastí). Později se zde mohou objevit i IR (je směrovač, který je připojený pouze do jedné oblasti a zprostředkovává směrování uvnitř této oblasti), ale to jen v případě, že by ABR směrovače nemohli být propojeny přímo mezi sebou. V ostatních oblastech budou tedy již zmiňované ABR a všechny zbylé směrovače typu IR (obr. 5).



Obr. 5: Rozdělení naší sítě do OSPF oblastí.
Směrovače označené popiskem Node B – E jsou typu ABR. Jsou to hlavní uzly (směrovače) na retranslačních místech B – E, které jsou od sebe vzdáleny stovky metrů až několik kilometrů. Směrovač Node A je v našem případě typu ASBR a poskytuje přímou cestu k internetové bráně. OSPF oblasti A – E jsou typu stub. Oblasti A1, A2 … jsou logické celky sítě obsahující pouze IR. Více k IP adresám použitým na obrázku v kapitole 6.

5.3 Autentizace OSPF

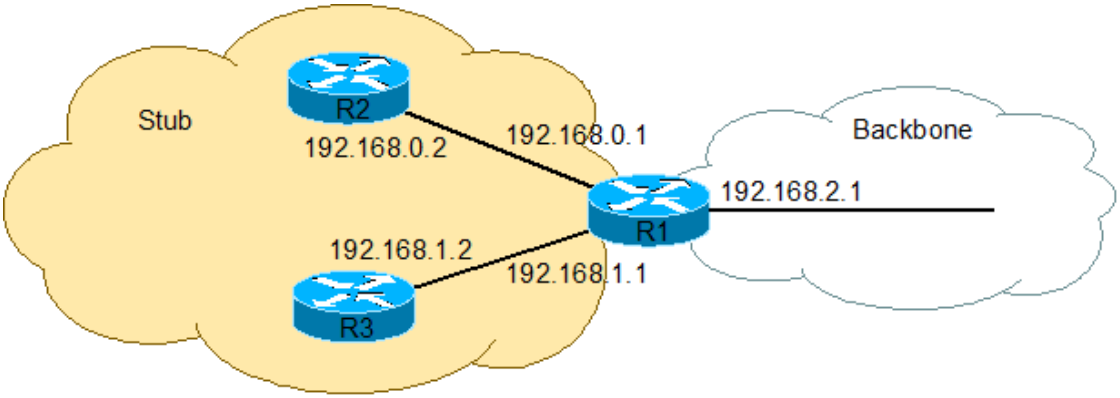
Ve výchozím nastavení je autentizace vypnutá, ale pokud chceme výměnu OSPF paketů zabezpečit, máme na výběr ze dvou typů zabezpečení.

Prvním typem zabezpečení je prostá autentizace, kterou se přenáší prostý text. Kdokoliv může odchytit paket a získat z něj snadno heslo. Tento typ zabezpečení se používá především pro zajištění přijetí správné konfigurace OSPF [1].

Druhým typem je MD5 zabezpečení, kterým se přenáší hash. Pomocí hashovaní funkce MD5 je z „authentication-key“, „key-id“ a obsahu OSPF paketu vytvořen hash a vložen do paketu. Hodnota „authentication-key“ není předávána po síti a pokud není nastavena je rovna „1“. Tento typ autentizace je nejčastěji používaný, protože nejen zabezpečuje správné přijetí konfigurace, ale také zabezpečuje přenos topologických informací [1].

5.4 Nastavení OSPF v RouterOS

Pro ukázku náročnosti nastavení OSPF na RouterOSu, ukážu jak nastavit tyto směrovače zapojené podle obrázku 6. Do vzorového příkladu jsem zakomponoval, téměř všechna nastavení, které budu poté provádět na směrovačích na naší síti.



Obr. 6: Oblast typu stub s hraničním směrovačem [2].

Oblast „backbone“ je typu default a oblast „stub“ je typu stub. Na oblasti stub použijeme pro autentizaci hashovaní funkci MD5. Předpokládáme, že všechny směrovače mají nainstalované a aktivované všechny potřebné balíčky, již jsou fyzicky propojeny a nastaveny IP adresy.

Na směrovači R1 prvně musíme vytvořit oblast, kterou nadefinujeme jako stub. Area id je unikátní označení oblasti. Toto id musí mít všechny směrovače v oblasti stejné. Název oblasti, může být na každém směrovači jiný. Je zde pouze pro přehlednost. V dalším kroku vložíme rozsahy IP adres připojených segmentů do patřičných oblastí (backbone, stub). Nakonec nastavíme jednotlivým rozhraním jejich autentizaci. V našem případě MD5 [2].

```
/routing ospf area add name=stub area-id=0.0.0.1 type=stub

/routing ospf network add network=192.168.2.0/24 area=backbone

/routing ospf network add network=192.168.1.0/24 area=stub
```

```
/routing ospf network add network=192.168.0.0/24 area=stub

/routing ospf interface add interface=ether1 authentication=md5
authentication-key=nasKlic authentication-key-id=2

/routing ospf interface add interface=ether2 authentication=md5
authentication-key=nasKlic authentication-key-id=2
```

Směrovač R2 nastavíme obdobně. Tedy vytvoříme OSPF oblast, poté ji přidělíme rozsahy IP adres připojených segmentů a nakonec na rozhraních nastavíme autentizaci, aby směrovače spolu komunikovaly [2].

```
/routing ospf area add name=stub area-id=0.0.0.1 type=stub

/routing ospf network add network=192.168.0.0/24 area=stub

/routing ospf interface add interface=ether1 authentication=md5
authentication-key=nasKlic authentication-key-id=2
```

Směrovač R3 má totožné nastavení jako směrovač R2. Jediné v čem se nastavení obou směrovačů bude lišit, jsou rozsahy IP adres připojených segmentů.

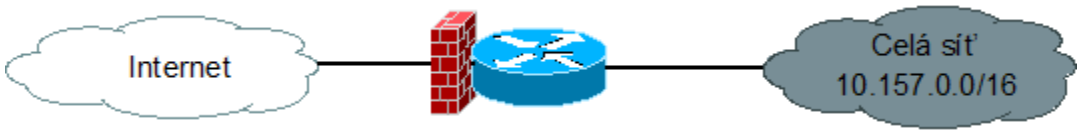
```
/routing ospf area add name=stub area-id=0.0.0.1 type=stub

/routing ospf network add network=192.168.1.0/24 area=stub

/routing ospf interface add interface=ether1 authentication=md5
authentication-key=nasKlic authentication-key-id=2
```

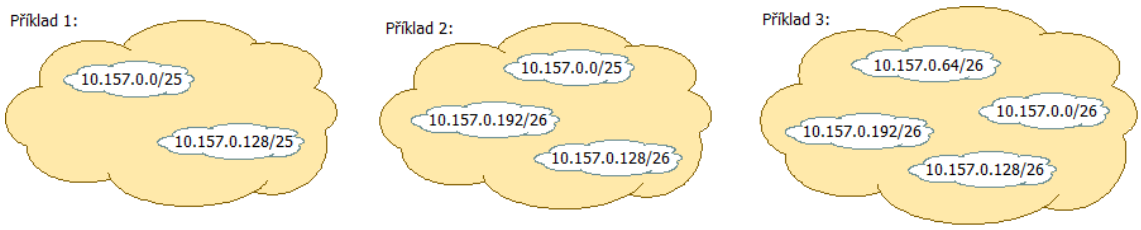
6 Rozdělení sítě na menší podsítě

Schéma sítě, kterou budeme dále dělit, máme na obrázku 7. Pro směrování je třeba rozdělit síť do menších celků podle potřeby IP adres. Nejlepším způsobem jak docílit efektivního rozdělení na menší logické celky je začít soupiskou potřebných adres na jednotlivé segmenty a postupně je sčítat.



Obr. 7: Schéma propojení metropolitní sítě s Internetem

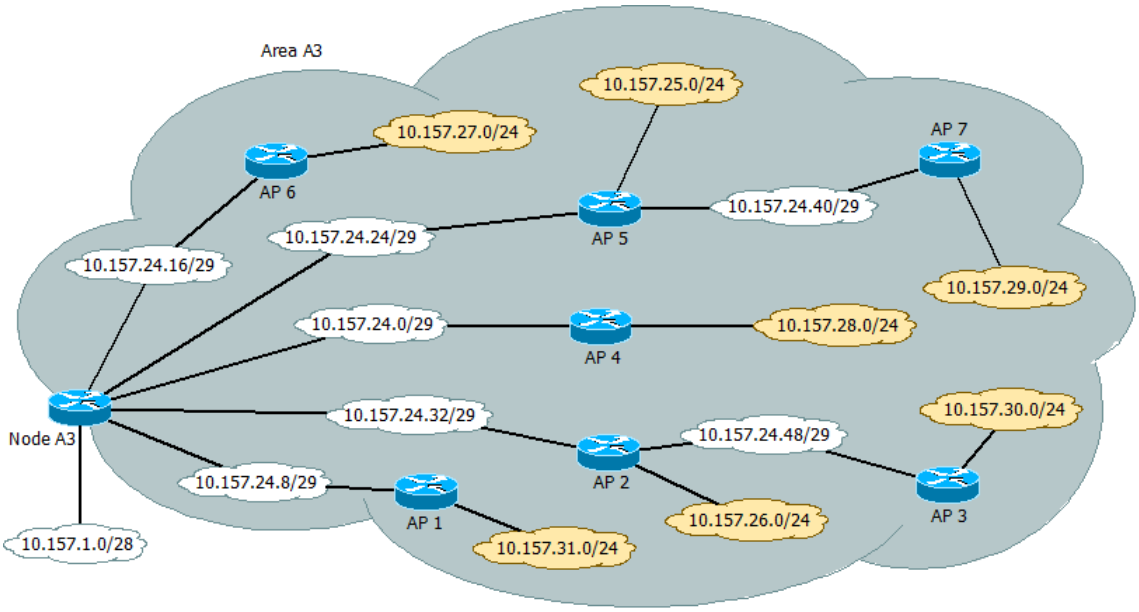
Pro koncový vysílač, na který se již připojují klienti, použiju rozsah o 256 adresách. Proč celé 1 C uvedu na příkladu. Náš nejvytíženější přístupový bod se skládá z 3 sektorových antén a 3 access pointů v režimu bridge. Na tento přístupovém bodu je připojeno 65 uživatelů a každý druhý uživatel požádal o přidělení 2. nebo 3. IP adresy. Přesněji je zde použito 65 IP adres pro access pointy uživatelů a 101 IP adres pro počítače uživatelů. Po výměně 3 access pointů na přístupovém bodě za směrovač s 3 bezdrátovými rozhraními přidělím každému rozhraní podsíť s maskou dlouhou 25 a 26 bitů (1x podsíť s maskou o délce 25 bitů a 2x podsíť s maskou o délce 26 bitů). Tím získáme dostatečně velké podsítě. Rozdělit 1 C na přístupovém bodě můžeme několika způsoby (obr. 8), ale všechny jsou závislé na počtu klientů připojených na přístupový bod a množství síťových rozhraní, které využíváme.



Obr. 8: Příklady úplného rozdělení 1 C

Přístupové body budou většinou připojeny přímo na retranslační místo, ale může se stát, že budou připojeny i přes několik bezdrátových spojů. Směrovač na retranslačním bodě má podle modelu Routerboardu 6 – 8 rozhraní (bez nadstaveb). Pokud předpokládáme připojení jednoho přístupového bodu na právě jedno rozhraní směrovače na retranslačním místě, pak je možné připojit minimálně 5 přístupových bodů (jedno rozhraní je potřeba pro připojení směrovače dále do sítě). Tomuto směrovače přidělíme rozsah s maskou o délce 21 bitů (celkem 8 C), kde 1 C je režie mezi směrovačem a přístupovými body a zbylých 7 C se rozdělí

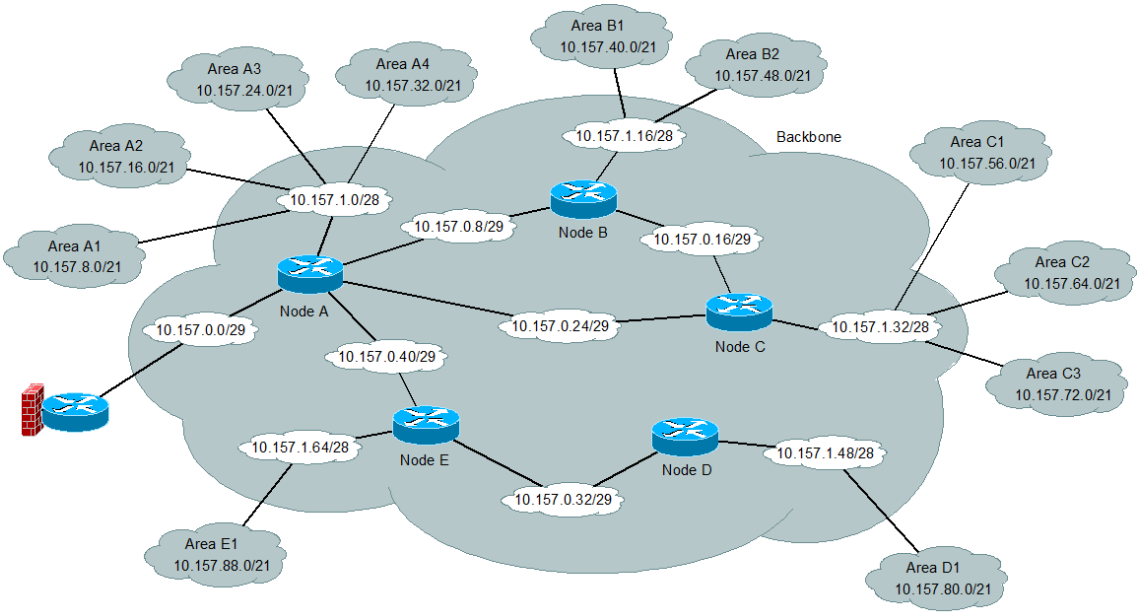
pro každý přístupový bod (obr. 9). Pokud bychom použil masku o délce 22 (celkem 4 C) bylo by to příliš málo a časem by se mohlo stát, že by jsme byli nuceni přidat nový rozsah. Naopak maska o délce 20 (celkem 16 C) je příliš velká a nevyužili by jsme celý tento rozsah.



Obr. 9: Detailní zobrazení podsítí za směrovačem na retranslačním bodě

Směrovače jsou navzájem propojeny rozsahy s maskou délky 29 bitů (8 IP adres), protože samotné spoje můžou být realizovány pomocí access pointů v režimu bridge, tudíž je potřeba 2 IP adres pro access pointy a 2 IP adres pro rozhraní směrovačů. Nesmíme zapomenout, že se jedná o přestavbu a musíme tedy počítat i s možností využití stávajících zařízení.

Na retranslačním místě nalezneme několik směrovačů zprostředkující komunikaci mezi páteří celé sítě a přístupovými body pro uživatele, ale jen jeden směrovač propojující retranslační body mezi sebou. Všechny směrovače jsou na retranslačním bodě propojeny switchem a jejich celkový počet bude 2 -5. Nesmíme zapomenout, že s každým směrovačem je spojeno okolo 5 spojů s přístupovým bodem tedy 5 antén, což při 3 směrovačích už dělá 15 směrových antén na jednom místě, což je poměrně hodně. Rozsah mezi těmito směrovači bude s maskou o délce 28 bitů (16 IP adres), protože další adresy můžeme využít pro servery (ftp server, smtp server, web server …), webkamery nebo pro servisní účely (připojení notebooku na místě). Výsledné schéma ukazuje obrázek 10.



Obr. 10: Detailnější zobrazení rozdělení sítě na páteř a podsítě páteřních uzlů

Na tomto obrázku je páteřní oblast rozkreslená dopodrobna. Směrovače označené velkým písmenkem jsou hlavní směrovače na síti (páteřní uzly). Jsou umístěné na retranslačních bodech. Jsou to hlavní uzly celé sítě, proto některé linky je lepší zálohovat proti výpadku (záleží na viditelnosti a dostupnosti dalších retranslačních stanic).

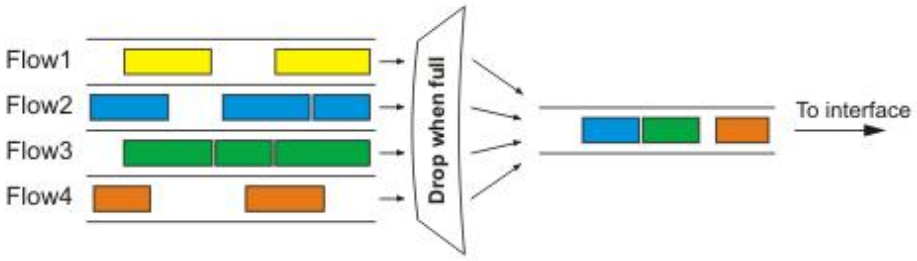
7 Rate-limiting

Nyní víme, jak bude síť vypadat i jak nastavit směrování. Dalším krokem je vybrat vhodné omezování rychlosti, tedy rate-limiting, a implementovat ho. Směrovače používají operační systém RouterOS, v němž se fronty pro omezení rychlosti nazývají „queues“. Jsou zde implementovány tyto mechanismy frontování:

- PFIFO - Packets First-In First-Out
- BFIFO - Bytes First-In First-Out
- SFQ - Stochastic Fairness Queuing
- RED - Random Early Detect
- PCQ - Per Connection Queue
- HTB - Hierarchical Token Bucket

7.1 PFIFO (Packets First-In First-Out)

Jak už název napovídá, tento mechanismus je založený na FIFO algoritmu. Tento algoritmus je také znám jako „fronta“. Paket, který první vstoupí do fronty, také z ní první vystoupí (obr. 11) Jediným parametrem, který se musí nastavit, je počet paketů, které se budou ukládat do fronty. Všechny pakety, které jsou nad rámec této fronty, jsou zahozeny [1].



Obr. 11: PFIFO/BFIFO, převzato z [1].

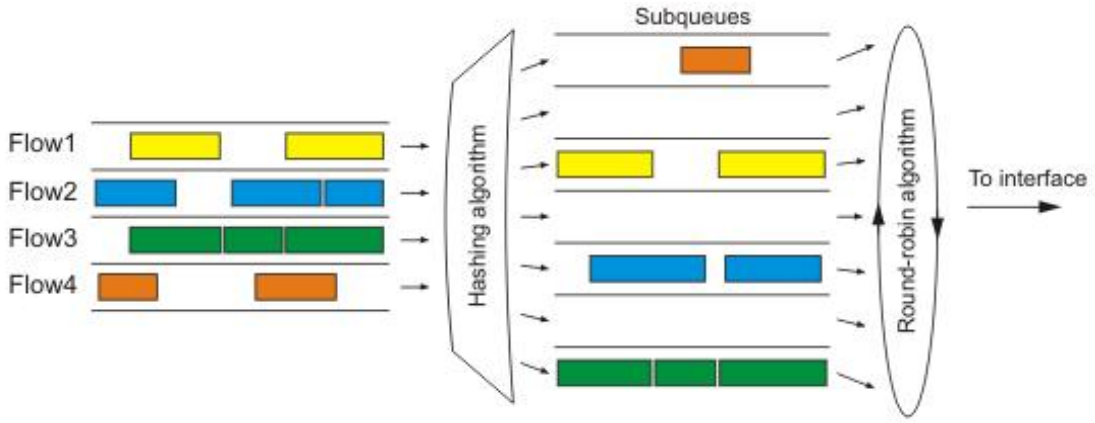
7.2 BFIFO (Bytes First-In First-Out)

Jak už název napovídá, jedná se o praktický stejný mechanismus jako PFIFO (obr. 11). Jediný rozdíl mezi nimi je v tom, že u PFIFO nastavujeme maximální počet paketů a u BFIFO nastavujeme počet Bytů. Vše nad rámec fronty, je opět zahozeno [1].

7.3 SFQ (Stochastic Fairness Queuing)

Tento mechanismus využívá hashovacího a round-robin algoritmu. Pomocí hashovaní funkce rozdělí provoz do menších FIFO subfront („subqueues“). Datový tok je hashován podle

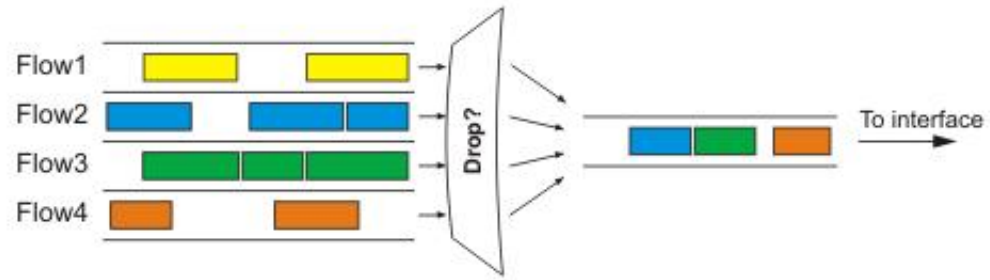
zdrojové a cílové adresy a portu. Pakety jsou poté díky round-robin algoritmu odebírány rovnoměrně z každé této subfronty (obr. 12). RouterOS má implementováno 1024 těchto subfront. Maximální počet front nelze nijak nastavit, ale je možné ovlivnit, po kolika sekundách se hashovaní funkce změní (atribut: sfq-perturb) a kolik bytů se odebere z fronty, když přijde na řadu (atribut: pcq-allot) [1].



Obr. 12: SFQ, převzato z [1]

7.4 RED (Random Early Detect)

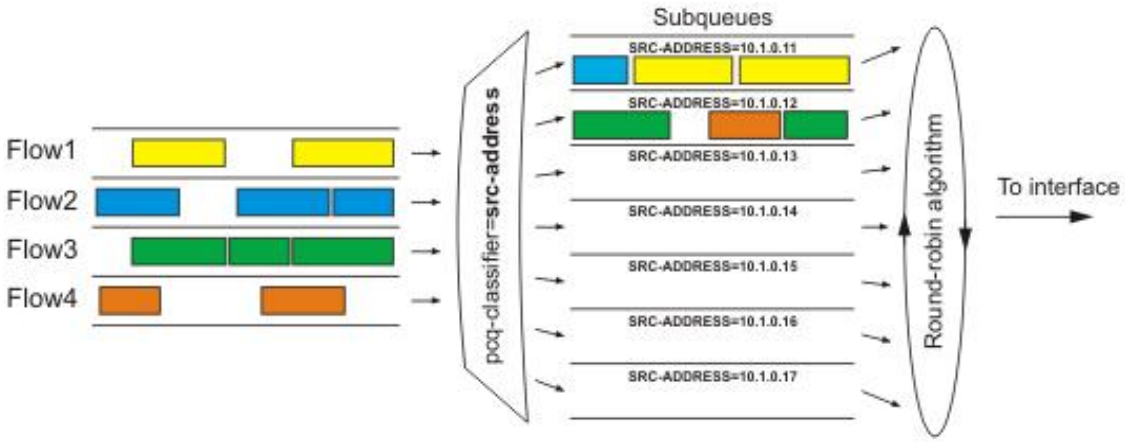
Tento mechanismus je někdy nazýván taky jako „Random Early Drop“, kvůli jeho vlastnostem. Fronta paketů se pomalu plní. Ve chvíli, kdy počet paketů přesáhne mezní hranici „red-min-threshold“, začne tento mechanismus náhodně zahazovat pakety. Čím více se počet paketů ve frontě blíží mezní hranici „red-max-threshold“, tím více náhodných paketů zahazuje (nárůst počtu zahozených paketů je lineární). Pokud přesáhne počet paketů ve frontě „red-max-threshold“, pak začne mechanismus zahazovat pakety tak dlouho, dokud počet paketů ve frontě neklesne pod počet udávaný „red-min-threshold“ (obr. 13). V RouterOSu lze nastavit délka fronty i obě mezní hranice [1].



Obr. 13: RED, převzato z [1].

7.5 PCQ (Per Connection Queue)

Tento mechanismus je velmi podobný SFQ. Tak jako SFQ vytváří menší subfronty („subqueues“), které plní postupně pakety. Na rozdíl od SFQ nepoužívá k tomu hashovaní funkci nýbrž klasifikátor (obr. 14). Tímto klasifikátorem může být zdrojová adresa, cílová adresa, zdrojový port, cílový port nebo libovolná jejich kombinace. Dalšími atributy, které můžeme nastavit je maximální počet paketů v subfrontě a maximální počet paketů celé fronty (všech „subqueue“) [1].



Obr 14: PCQ s klasifikátorem zdrojové adresy, převzato z [1].

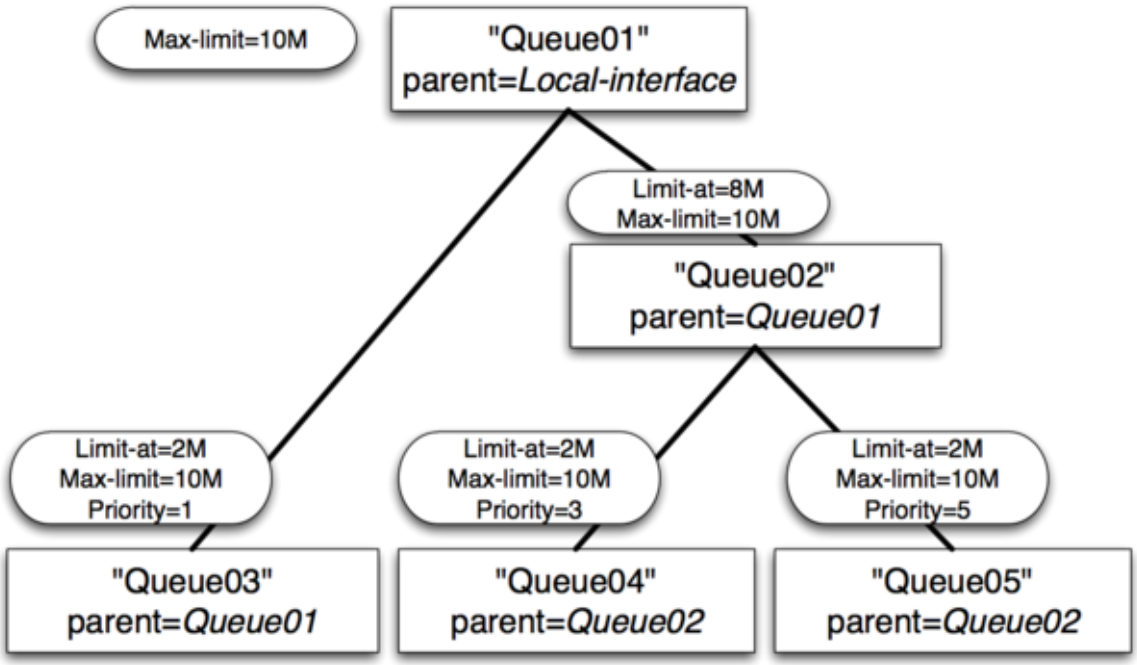
7.6 HTB (Hierarchical Token Bucket)

Tento mechanismus nám dovolí vytvořit hierarchickou strukturu nám již známých front (PFIFO, BFIFO, SFQ, RED a PCQ) a vytvořit vazby mezi nimi. Fronty v této struktuře se dělí na vnitřní fronty („inner“) a listy („leaf“). Vnitřní fronty jsou zodpovědné za rozdělení provozu a samotné listy frontují provoz. Rozdělování provozu se řídí 3 ukazateli [1]:

- *limit-at* – je to šířka pásma, kterou fronta vždy získá
- *max-limit* – je maximální šířka pásma, která frontou proteče
- *priority* – priorita určuje, která fronta dostane zbývající konektivitu po rozdělení *limit-at* všem frontám, pokud těchto front je více, tak je rozdělí rovnoměrně

Vysvětlení na příkladu. Máme konektivitu 10 Mbitů a 3 uživatele A, B a C s tarifem 10Mbitů s agregací 1:5. Pro demonstraci HTB každému uživateli dáme jinou prioritu a přidáme zde vnitřní frontu „Queue02“ s hodnotami *limit-at* = 8 Mbitů a *max-limit* = 10 Mbitů. Uživateli A je přidělena fronta „Queue03“ s *priority* = 1, uživateli B je přidělena „Queue04“ s *priority* = 3

a uživateli C je přidělena „Queue05“ s *priority* = 5. Všechny fronty typu list mají hodnoty *limit-at* = 2 Mbity a *max-limit* = 10 Mbitů. Schéma nastavení front je na obrázku 15.



Obr. 15: Schéma příkladu HTB

Pokud bude stahovat právě jeden uživatel, bude mu přiděleno celých 10 Mbitů.

Pokud bude stahovat uživatel A i uživatel B, budou uživateli A přiděleny 2 Mbity a uživatel B dostane 8Mbitů.

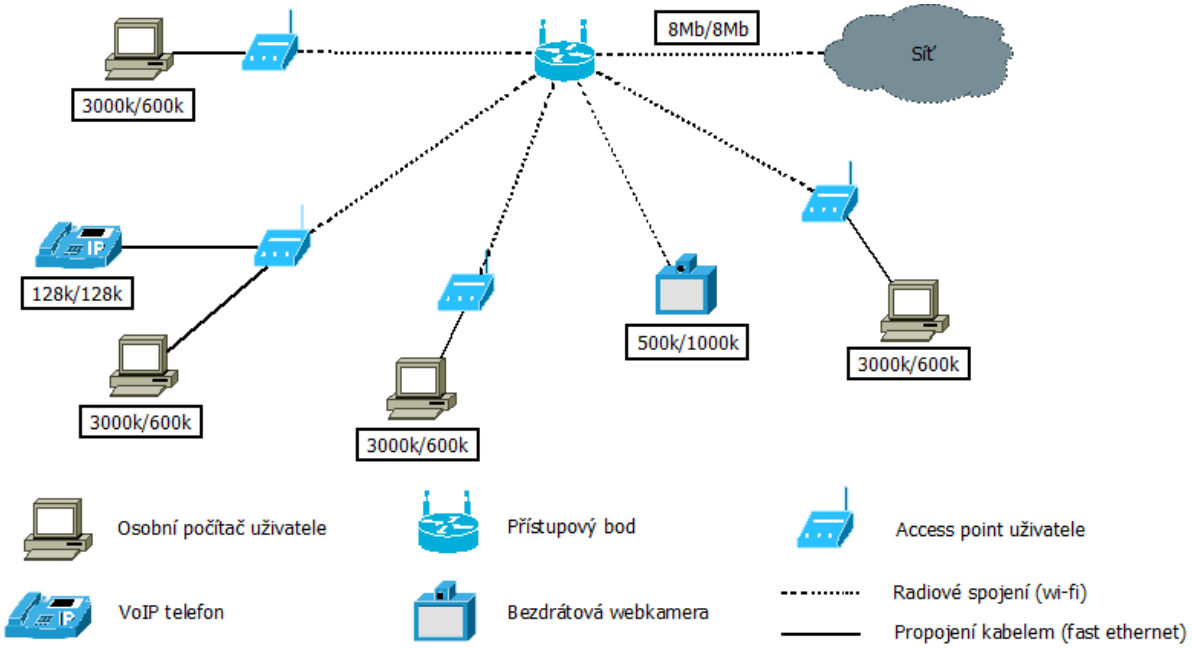
Pokud budou stahovat všichni 3 uživatelé, pak uživatel A dostane 2 Mbity, uživatel B dostane 6 Mbitů a uživatel C dostane 2 Mbity, protože po naplnění hodnot *limit-at* zbylá konektivita je přidělena podle priority.

7.7 Realizace

Pro naši síť jsem vybral HTB mechanismus s PCQ frontami aplikovanými na výstupní rozhraní. Rozhodl jsem se pro tuto variantu, protože díky HTB můžeme nejen omezit datový tok u uživatele, ale i zaručit některým službám dostatečnou šířku pásma (garantovat připojení pro VoIP, web kamery …). PCQ fronta zase zaručí rovnoměrné rozdělení konektivity mezi uživateli. Při realizaci bude třeba jinak nastavit směrovač na přístupovém bodě pro koncové uživatele a jinak směrovač na páteři.

Nastavení přístupových bodů pro koncové uživatele je složitější, proto samotné nastavení popíšu podrobněji na příkladu (obr. 16). Máme na přístupovém bodu 4 uživatele

s rychlostí 3 Mb download a 600 kb upload s agregací 1:6. Jeden z těchto uživatelů má navíc VoIP telefon s vlastní IP adresou, který vyžaduje 128 kb download a 128 kb upload. Na tento přístupový bod je také připojena naše bezdrátová webkamera, která potřebuje pro svůj provoz garantovaný přenos 1 Mb upload a 500 kb download. Přístupový bod je připojen bezdrátovým spojem o maximálním průtoku 8 Mb download a 8 Mb upload.



Obr. 16: Znázornění realizovaného příkladu

Jako první věc vytvoříme address listy pro každé zařízení/uživatele. V menu *IP -> Firewall -> Address Lists* vytvoříme nový seznam („list“) a vložíme do něj všechny IP adresy uživatele. Je vhodné vytvářet seznamy adres místo přímého vkládání jednotlivých adres, pro přehlednost a snadnější budoucí editaci.

Příklad:

```
/ip firewall address-list add list=uzivatel1-list address=10.0.0.0/29

/ip firewall address-list add list=uzivatel2-list address=10.0.0.2-
10.0.0.5

/ip firewall address-list add list=uzivatel3-list address=10.0.0.6,
10.0.0.7

/ip firewall address-list add list=uzivatel3_voip-list
address=10.0.0.9
```

Po vytvoření seznamů adres je třeba vytvořit označení paketů. V menu *IP -> Firewall -> Mangle* přidáme nová pravidla. Budou celkem 3 pro každý seznam adres. První pravidlo označí celý provoz, další dvě pravidla budou značit pakety downloadu a uploadu. U všech pravidel bude *chain* nastavený na *forward*, aby se pravidla vykonávala při průchodu paketu směrovačem. Pro pravidla značící celý provoz vložíme náš seznam adres do atributu *src-address-list*, protože komunikace začíná vždy u uživatele. Pro značení paketů vložíme náš seznam adres do atributu *src-address-list* pro pakety uploadu a do atributu *dst-address-list* pro pakety downloadu.

Příklad:

```
/ip firewall mangle add chain=forward src-address-list=uzivatel1_list
action=mark-connection new-connection-mark=uzivatel1_connection
passthrough=yes

/ip firewall mangle add chain=forward connection-
mark=uzivatel1_connection dst-address-list=uzivatel1_list action=mark-
packet new-packet-mark=uzivatel1_down passthrough=yes

/ip firewall mangle add chain=forward connection-
mark=uzivatel1_connection src-address-list=uzivatel1_list action=mark-
packet new-packet-mark=uzivatel1_up passthrough=yes
```

Nyní máme hotova pravidla pro označení provozu, proto vytvoříme PCQ fronty pro download a upload. V menu *Queue -> Type* vytvoříme 2 nové typy. Oba budou PCQ fronty a jako klasifikátor použijeme zdrojovou a cílovou adresu.

Příklad:

```
/queue type add name=pcq_down kind=pcq pcq-classifier=dst-address

/queue type add name=pcq_up kind=pcq pcq-classifier=src-address
```

Teď, když máme vytvořené seznamy adres, označený provoz a vytvořené fronty, je načase vytvářet HTB strukturu. Prvně vytvoříme 2 hlavní fronty reprezentující celkový download a upload. Obě tyto fronty budou mít za svého rodiče („*parent*“) *global-out*. Tím dosáhneme aplikaci omezení na všechny rozhraní. Směrovač na přístupovém bodu má obvykle 3 rozhraní a někdy se stalo, že jsme přidali další rozhraní (přidáním bezdrátové karty do volného miniPCI slotu). Při takovéto změně není třeba nic změnit, protože virtuální rozhraní *global-out* obsáhne i nové rozhraní automaticky. Takže pakety od uživatele budou omezeny,

ať už jdou do Internetu nebo k jinému uživateli v síti. Prioritu nastavovat nemusíme, ale pro přehlednost ji můžeme zadat.

Jelikož je přístupový bod připojen bezdrátově, je vhodné kromě hodnoty *max-limit* nastavit také hodnotu *limit-at* na polovinu maximální přenosové rychlosti. Tímto nastavením ošetříme problém v případě zarušení bezdrátového spoje. Kdybychom nenastavili hodnotu *limit-at*, tak v případě zarušení, by maximální průtok mohl poklesnout i o více než 1Mb, ale směrovač by se snažil stále rozdělovat celou hodnotu 8Mbitů. V důsledku toho by někteří uživateli dostali šířku pásma rovné hodnotě *limit-at* a někteří by dosahovali až hodnot *max-limit*. Tímto způsobem nevyřešíme problém zarušení, ale zajistíme, aby i v tomto případě se dělila maximální šířka pásma spravedlivě mezi uživateli.

Příklad:

```
queue tree add name=down parent=global-out queue=pcq_down max-limit=8M
limit-at=4M

queue tree add name=up parent=global-out queue=pcq_up max-limit=8M
limit-at=4M
```

Stejným pravidlem začneme vkládat jednotlivé uživatele/zařízení. Nesmíme zapomenout dát všem stejnou. Výjimkou jsou zařízení, u kterých chceme garantovat konektivitu. Tím jsou myšlena zařízení, která mají mít přednost všemi ostatními. Pro příklad access pointy a směrovače uživatelů. Čas od času je třeba se do těchto zařízení vzdáleně dostat a to i v době kdy je linka zatížená na maximum. Proto je zde vhodné pozvednout prioritu ještě o něco výše a tím upřednostnit datový tok zařízení. Pokud chceme garantovat rychlost, tedy agregaci 1:1, přidělíme atributům *max-limit* a *limit-at* stejnou hodnotu. Garantování rychlosti využijeme například i u zařízení, u kterých víme, jakou mají režii.

Příklad:

```
queue tree add name=uzivatel1_down parent=down packet-
mark=uzivatel1_down priority=6 max-limit=3M limit-at=500k

queue tree add name=uzivatel1_up parent=up packet-mark=uzivatel1_up
priority=6 max-limit=600k limit-at=100k

queue tree add name=uzivatel3_voip_down parent=down packet-
mark=uzivatel3_voip_down priority=5 max-limit=128k limit-at=128k

queue tree add name=uzivatel3_voip_up parent=up packet-
mark=uzivatel3_voip_up priority=5 max-limit=128k limit-at=128k
```

```
queue tree add name=webkamera1_down parent=down packet-
mark=webkamera1_down priority=5 max-limit=500k limit-at=500k

queue tree add name=webkamera1_up parent=up packet-mark=webkamera1_up
priority=5 max-limit=1M limit-at=1M
```

8 Skripty v RouterOS

Skripty používáme pro ulehčení práce popřípadě pro automatizaci některých činností. Tvoříme je z příkazů, které používáme v CLI. Navíc pro různé výpočty je zde několik operátorů, kteří nám mohou ulehčit práci. Mezi ty základní patří plus, mínus, krát, děleno, logický součet, logický součin, logický zápor, bitová inverze a mnoho dalších.

Každý řádek skriptu je jeden příkaz, který se má vykonat. Pokud chceme, aby příkaz pokračoval na dalším řádku (třeba pro přehlednost), musíme na konec prvního řádku vložit „\“ [1].

```
:if ([/ping $ip count=10] = 0) do { ...
```

Tento příkaz můžeme tedy napsat na dva řádky takto:

```
:if ([/ping $ip count=10] = 0) \

do { ...
```

Každý příkaz začínající „/“ je nastavení nebo funkce směrovače.

```
/ping 192.168.1.7

/routing ospf network add network=192.168.1.0/24 area=stub
```

V prvním příkladě se jedná o funkci směrovače. Zjišťujeme odezvu k IP adrese 192.168.1.7. V druhém případě přidáváme do menu routing, ospf, network novou síť 192.168.1.0/24 k oblasti se jménem stub.

Každý příkaz začínající „:“ je funkce systému. Jedná se například o funkce jako je ukládání proměnné do paměti, cykly, podmínka a mnoho dalších. Tyto funkce můžou být i složené, pak se tedy dává „:“ pouze před první příkaz [1].

```
:if ([/ping $ip count=10] = 0) do {…}
```

Pokud adresa 192.168.1.7 desetkrát neodpoví, provede se blok příkazů ve složených závorkách. Jak je patrno na příkladu, před „do“ se dvojtečka nepíše.

Při tvorbě skriptů určitě budeme potřebovat ukládat do paměti a opět z ní načíst. RouterOS disponuje několika typy proměnných. Jsou to string, „boolean, number, time interval, IP address, internal number a list. Při definici proměnné nedefinujeme její typ. To dělá systém za nás sám. Definujeme pouze, zda je proměnná global nebo local. Tedy jestli je dostupná pro všechny skripty do restartu směrovače (tedy global) anebo je proměnná dostupná pouze v našem skriptu. Po skončení skriptu se smažou z paměti všechny proměnné typu local [1]. Deklarovat proměnou můžeme navíc dvěma způsoby. Pro příklad nadefinuji lokální proměnnou „x“ s hodnotou „test“.

```
:local x „test“
```

Nebo také druhým

```
:set x „test“
```

V případě, že systém špatně rozpoznal typ naší proměnné nebo pokud potřebujeme převést proměnou na jiný typ, jsou v systému implementovány funkce toarray, tobool, toid, toip, tonum, tostr a totime. Myslím si, že tyto funkce příliš nevyužijete, protože RouterOS před provedení operace sám převede proměnou na patřičný typ [1]. Pro příklad vytvoříme dva řetězce „x1“ s hodnotou „12“ a „x2“ s hodnotou „34“. Sečteme obě ty to čísla a vypíšeme na obrazovku. To samé provedeme ještě jednou, ale prvním znakem z každého řetězce.

```
:set x1 "12"

:set x2 "34"
```

Prvně jsme si nadefinovali proměnné x1 a x2. Pro zkontrolování obsahů našich proměnných použijeme příkaz environment.

```
:environment print

Global Variables
Local Variables
x1=12
x2=34
```

Teď naše proměnné sečteme jako by to byly čísla a vypíšeme na obrazovku. Výpis provedeme pomocí příkazu put.

```
:put ($x1 + $x2)

46

:put ([:pick $x1 0] + [:pick $x2 0])

4
```

Jak již je patrno z výše uvedeného příkladu, pro volání proměnné dáme před proměnnou tento znak „$“. Každý příkaz musí začínat „:“ nebo „/“, proto cokoliv bez těchto symbolů je výpis z konzole.

8.1 Cykly ve skriptech

Pro tvorbu cyklu můžeme využit jeden ze 4 typu cyklů. Jsou to do, for, foreach, while.

:do {co se má provést} while (podmínka) – Toto je cyklus s podmínkou na konci. Provádí se tak dlouho, dokud podmínka platí [1]. Pro příklad proměnné „x“ dáme hodnotu 1. Budeme ji postupně přičítat 1, dokud nedosáhneme čísla 4.

```
:set x 1

:environment print

Global Variables
Local Variables
x=1

:do {:set x ($x + 1)} while ($x < 4)

:environment print

Global Variables
Local Variables
x=4
```

:for x from=1 to=100 step=5 do={co se má provést} – Druhý cyklus v pořadí má pevný počet opakování. „x“ je jméno proměnně, kterou budeme inkrementovat. Pokud chceme dekrementovat, stačí zvolit za „step“ záporné číslo. From je počáteční hodnota proměnné „x“. To je naopak konečná hodnota proměnné „x“. Step udává, o kolik se má inkrementovat

proměnná „x“ [1]. Pro příklad jsem zvolil proměnnou „y“ s počáteční hodnotou 0. Budeme přičítat 2 do té doby, dokud „y“nebude 4.

```
:for y from=0 to=4 step=2 do={:put $y}
```

:foreach x in=list do={ co se má provést } – Třetí cyklus je zjednodušený cyklus for pro průchod všemi prvky seznamu (listu). „x“ je index právě procházeného seznamu. „list“ je jméno seznamu, který budeme celý procházet [1]. Níže uvedené příkazy jsou ekvivalentní.

```
:for x from=0 to([:len $list] - 1) step=1 do={(:put "Hellou World")}
```

```
:foreach x in=list do={(:put "Hellou World")}
```

:while (podmínka) do={ co se má provést } – Poslední je cyklus s podmínkou na začátku. Pokud podmínka je splněna, provede se cyklus [1]. Pro příklad proměnné „x“ dáme hodnotu 1. Budeme ji postupně přičítat 1, dokud nedosáhneme čísla 4.

```
:set x 1

:environment print

Global Variables
Local Variables
x=1

:while ($x < 4) do {:set x ($x + 1)}

:environment print

Global Variables
Local Variables
x=4
```

8.2 Další funkce

Neměl bych opomenout další důležité funkce, které se nám můžou hodit při tvorbě skriptů.

:resolve (doména) - Přeloží doménu na IP adresu, ale jen v případě, že jsou korektně nastaveny DNS servery ve směrovači [1].

: put (text) – Vypíše řetězec do terminálu [1].

:time [funkce] - Vrátí čas, který trvalo vykonat funkci [1].

:pick [(proměnná) (počáteční index) (konečný index)] – je funkce pro práci s poli, listy … Pokud vyplníme jen parametr „proměnná“ pak vrací hodnotu prvního prvku v poli. Pokud vyplníme i „počáteční index“, vrátí hodnotu na indexu v poli. Pokud vyplníme všechny 3 parametry, pak vrátí pole prvků, jež začíná v původním poli na indexu od „počátečního indexu“ a končí na „konečném indexu“ kromě posledního prvku. Hodnota na pozici „konečný index“ se do nového pole nepřidá [1].

:log (typ) (text) – Tato funkce zapíše do logu daného typu (například „info“, „error“…) text, který chceme předat. Je potřeba vyplnit oba parametry [1].

:len (proměnná) – Vrátí délku pole, listu … [1].

:if ((podmínka)) do={ (platí) } else={ (neplatí) } – Pokud „podmínka“ platí, pak se vykoná obsah složených závorek za příkazem „do“, pokud neplatí tak se vykoná obsah složených závorek za příkazem „else“ [1].

:enviroment print – Vypíše do terminálu všechny proměnné uložené v paměti tedy globální i lokální [1].

8.3 Práva skriptu

Při tvorbě skriptu musíme zadat, jaká práva („policy“) skript bude mít. Práva se odvíjejí od práv uživatelů. Tyto práva omezují možnosti, které skript může vykonávat. Práva skriptu přiděluje uživatel, který jej vytvořil, ale maximálně taková která on sám má. Tedy práva jsou zde proto, aby uživatel nemohl vytvořit skript, který bude provádět činnost, kterou uživatel sám dělat nemůže. Pro přehlednost jsem všechny práva popsal, co uživateli umožní. Máme na výběr z těchto možností (lze je libovolně kombinovat) [1]:

- ftp – Uživatel se může nalogovat přes ftp do směrovače a manipulovat s daty (stáhnout, nahrát).

- local – Uživatel se může nalogovat se do terminálu směrovače.

- policy – Uživatel má možnost upravovat nastavení uživatelů, vytvářet a mazat účty a skupiny uživatelů.

- read – Uživatel může číst nastavení ze směrovače.

- reboot – Uživatel má právo restartovat směrovač.

- ssh – Uživatel se může nalogovat přes SSH do směrovače.

- telnet – Uživatel se může nalogovat přes Telnet.

- test – Uživatel má práva pro používání funkcí ping, traceroute, bandwith test ve směrovači.

- web – Uživatel se může nalogovat přes webové rozhraní do směrovače.

- write – Uživatel má možnost číst i zapisovat nastavení ve směrovači.

8.4 Možností spuštění skriptu

Pokud máme napsaný skript, nastavíme mu patřičná práva, už chybí jen ho spustit. Spuštění můžeme provést několika způsoby. První způsob jak spustit náš skript je ručně v grafickém rozhraní programu Winbox. Druhou možností je spuštění pomocí druhého skriptu, který náš skript volá (odkazuje na něj, aby se spustil). Poslední možností je reakce na událost způsobenou časovačem („scheduler“), monitoringem („netwatch“) nebo VRRP [3].

8.5 Příklady skriptů

8.5.1 Restart rozhraní

Již několikrát jsem na naší síti zaznamenal, že přes bezdrátový spoj přestaly protékat data i přesto, že byl ve stavu „connected“. Ihned po tom co jsem restartoval bezdrátové rozhraní (vypnul a zapnul) vzdáleně přes SSH, data začala zase protékat. Abych urychlil reakci na výpadek datových služeb, vytvořil jsem jednoduchý skript, který se spouští každou minutu a testuje 10x odezvu protější strany bezdrátového spoje. Pokud protější strana neodpoví alespoň jednou, restartuje se bezdrátové rozhraní a do logu směrovače se zapíše, že byl proveden restart rozhraní.

Na začátku si musíme nadefinovat proměnné, které budeme používat. V našem případě je to jméno bezdrátového rozhraní, které budeme restartovat a IP adresa protějšího zařízení.

```
:local ip "10.157.128.6"

:local rozhrani wlan_backbone
```

Dalším krokem je podmínka. Naším cílem je po 10 ztracených paketech restartovat rozhraní. Příkaz ping má parametry, a proto jej musíme dat do hranatých závorek. To platí pro všechny příkazy s atributy.

```
:if ([/ping $ip count=10] = 0) do {}
```

A nakonec doplníme náš skript o restart. Ten složíme ze dvou příkazů. Jeden pro vypnutí rozhraní a druhý pro zapnutí rozhraní. Bohužel pokud by směrovač tyto příkazy vykonal za sebou bez pauzy, nemuselo by se nic stát. Proto přidám ještě jeden řádek, aby chvíli počkal. Nakonec přidáme ještě zápis do logu, kde bude napsáno, že skript restartoval rozhraní. Do budoucna se stačí podívat do logu a uvidíme tam čas, kdy se to stalo a jak často se to opakovalo.

```
/interface wireless disable $rozhrani

:delay 1

/interface wireless enable $rozhrani

:log error ("Rozhrani wlan_backbone bylo restartovano Skriptem")
```

Takto napsaný skript stačí vložit do RouterOS přes FTP nebo WinBox a nastavit spouštění každou minutu.

Celý skript i s komentáři je uveden v příloze 2.

8.5.2 Zavedení rate-limitingu

Skript zavedení rate-limitingu má ulehčit práci s nastavováním nově nasazeného směrovače jako přístupového bodu. Spustíme ho tedy jen jednou pro vložení daných pravidel. Dále již není potřeba, proto nevyužíváme služby časovače. Realizaci provádíme podle kapitoly 7.7.

Na začátku si nastavíme všechny potřebné proměnné. V tomto případě se tedy jedná o hodnotu maximálního downloadu a maximálního uploadu. Hodnoty uploadu a downloadu udáváme v bitech. Poté spustíme skript, který přidá do směrovače:

- 2 typy front („pcq-download“ a „pcq-upload“)
- 2 fronty na výstup („download“ a „upload“), jejíž „limit-at“ bude roven polovině „max-limit“
- 3 pravidla značení paketů pro zařízení klientů (access pointy v režimu bridge, které jsou využívány pro připojení na přístupový bod)
- 2 fronty pro zajištění přístupu ke klientským zařízením („limit-at“ = 128 k a „max-limit“ = 256 k)

Celý skript i s komentáři je uveden v příloze 3.

8.5.3 Přidělení rychlosti uživateli

Pro použití skriptu pro přidělení rychlosti uživateli potřebujeme, aby ve směrovači byly již implementovány fronty „download“ a „upload“. Tyto fronty včetně značení datových toků můžeme vytvořit sami ručně nebo pomocí skriptu pro zavedení rate-limitingu (viz kapitola 7.2.2). Pro omezení rychlosti uživatele nám stačí vytvořit 3 pravidla značení paketů pro označení toku dat daného uživatele a 2 fronty pro samotné omezování. Abychom jednoznačně odlišili jednotlivé uživatele, použijeme jejich jméno, příjmení, oblast kde bydlí a vysílač, na který se připojují. Pomocí všech těchto údajů vytvoříme jednoznačné názvy pravidel v RouterOSu. Dále budeme potřebovat IP adresy jejich domácích počítačů a access pointů v režimu bridge.

```
:local ip "10.157.143.98"

:local ap "10.157.143.99"

:local jmeno david

:local prijmeni balcarek

:local oblast centrum

:local node billa
```

Všechny výše uvedené proměnné nastavíme před spuštěním skriptu. Protože je příliš zdlouhavé a nepřehlední stále dokola vkládat do atributů dlouhé řetězce s názvem pravidel a seznamů, vytvoříme si ve skriptu několik pomocných proměnných. Vytvoříme je z uživatelem definovaných proměnných pomocí funkce spojování řetězců. Operátor pro operaci spojení 2 řetězců je tečka. Jedná se o jméno seznamu IP adres uživatele, jména pravidel pro značení datových toků a jména front uživatele.

```
:local list ($oblast . "-" . $node . "-" . $prijmeni . "-" .
$jmeno . "-list")

:local connection ($oblast . "-" . $node . "-" . $prijmeni . "-"
. $jmeno . "-connection")

:local down ($oblast . "-" . $node . "-" . $prijmeni . "-" .
$jmeno . "-down")

:local up ($oblast . "-" . $node . "-" . $prijmeni . "-" .
$jmeno . "-up")
```

Obsah pomocných proměnných je dle zadání výše:

- „list“ = „billa-centrum-balcarek-david-list“
- „connection“ = „billa-centrum-balcarek-david-connection“
- „down“ = „billa-centrum-balcarek-david-down“
- „up“ = „billa-centrum-balcarek-david-up“

Největším problémem bylo vkládání IP adres do address listu, přesněji načítání z proměnné. V případě, že vložíme do proměnných pouze 1 IP adresu, systém ji bere jako proměnnou typu IP address. Ale vložíme-li více adres, systém pracuje s polem. Problém jsem vyřešil podmínkou, která spočítá počet míst v poli/řetězci. Protože operujeme s rozsahem 10.157.0.0/16, nejmenší počet míst řetězce (jedné IP adresy) je minimálně 10, maximální pak 14. Pokud systém zjistí, že proměnná má 10 -14 znaků/objektů, pole počítá s jednou IP adresou. V případě, že proměnná má jiný počet znaků/ objektů pole, systém počítá s polem IP adres a vkládá je postupně do „address list“.

```
:if ([:len $ip] = 10 || [:len $ip] = 11 || [:len $ip] = 12 ||
[:len $ip] = 13 || [:len $ip] = 14) do {<< pro 1 IP >>} else {<< pro
vice IP >>}
```

Celý skript i s komentáři je uveden v příloze 4.

9 Závěr

Cílem této práce bylo navrhnout metropolitní síť na 3. vrstvě OSI modelu vycházející z reálné radiové sítě realizované na 2. vrstvě OSI modelu. Začalo se vybráním vhodné platformy pro směrovač. Nejvhodnější pro naše využití byl Routerboard s operačním systémem RouterOS. S tímto operačním systémem jsem do té doby neměl žádné zkušenosti. Po bližším seznámení se směrovačem a jeho operačním systémem jsem návrh sítě obohatil o příklady implementace OSPF oblastí, implementaci rate-limitingu a několik skriptů usnadňující implementaci rate-limitingu.

K původní síť na 2. vrstvě bylo připojeno 380 uživatelů, celková konektivita do Internetu byla 25 Mb/s / 25Mb/s a uživatelé měli tarif 3 Mb/s / 500 kb/s s agregací 1:10. Průměrná rychlost download uživatele byla 1,2 Mb/s – 1,8 Mb/s (tab. 1). Po přestavbě sítě na 3. vrstvu OSI modelu byl počet uživatelů 450, celková konektivita 25 Mb/s / 25Mb/s a uživatelé měli stejný tarif 3 Mb/s / 500 kb/s s agregací 1:10. I přesto, že během přestavby přibylo do sítě 70 nových uživatelů, průměrná rychlost download uživatele vzrostla na 1,5 Mb/s – 2,6 Mb/s (příloha. 5).

K takovému zlepšení přispěla nejen přestavba na 3. vrstvu OSI modelu, ale i vhodná implementace rate-limitingu. Během své práce jsem získal spoustu pozitivních zkušeností především v práci s operačním systémem RouterOS.

10 Literatura

[1] *MikroTik Routers and Wireless* [online]. Document revision: 3.40. 2005, September 26, 2007. MikroTik RouterOS V2.9 Reference Manual. Dostupné z WWW: <http://www.mikrotik.com/testdocs/ros/2.9/>.

[2] *Manual:OSPF Case Studies - MikroTik Wiki* [online]. 2009, modified on 14 April 2010. MikroTik Wiki. Dostupné z WWW: <http://wiki.mikrotik.com/wiki/Manual:OSPF_Case_Studies>.

[3] *Manual:Scripting - MikroTik Wiki* [online]. 2009, modified on 26 April 2010. MikroTik Wiki. Dostupné z WWW: <http://wiki.mikrotik.com/wiki/Scripting>.

11 Přílohy

I. Obrázek zařízení společnosti MikroTik Routerboard 433AH, převzato z http://www.routerboard.com/comparison.html
II. Skript restart rozhraní
III. Skript zavedení rate-limitingu
IV. Skript přidání rychlosti uživateli
V. Tabulka: Měření rychlosti download u uživatelů po přestavbě sítě.